L Number	Hits	Search Text	DB	Time stamp

1	155	(hash with join$3) and (@ad<20010426)	USPAT;
			US-PGPUB;
			IBM_TDB
				2004/04/22
				13:06

2	90	(hash with join$3) and (aggregat$5 or (intermed$9 with result)) and (@ad<20010426)	USPAT;
			US-PGPUB;
			IBM_TDB
				2004/04/22
				13:14

6	45	(hash with join$3) and (((online or (on adj line)) adj aggregat$5) or (intermed$9 with result)) and (@ad<20010426)	USPAT;
			US-PGPUB;
			IBM_TDB
				2004/04/22
				13:18

7	0	(hash with join$3) and (split adj vector) and (((online or (on adj line)) adj aggregat$5) or (intermed$9 with result)) and (@ad<20010426)	USPAT;
			US-PGPUB;
			IBM_TDB
				2004/04/22
				13:19

9	0	(hash with join$3) and (split adj vector) and (@ad<20010426)	USPAT;
			US-PGPUB;
			IBM_TDB
				2004/04/22
				13:19

10	15	(hash with join$3) and vector and (((online or (on adj line)) adj aggregat$5) or (intermed$9 with result)) and (@ad<20010426)	USPAT;
			US-PGPUB;
			IBM_TDB
				2004/04/22
				13:23

12	41	(hash with join$3) and vector and (@ad<20010426)	USPAT;
			US-PGPUB;
			IBM_TDB
				2004/04/22
				14:24

-	2	(parallel$5 with hash$3 with (join$3 or merg$3 or combin$5)).ab. and (@ad<20010426)	USPAT;
			US-PGPUB;
			IBM_TDB
				2003/05/16
				12:54

-	78	(hash$3 with (join$3 or merg$3 or combin$5)).ab. and (@ad<20010426)	USPAT;
			US-PGPUB;
			IBM_TDB
				2004/04/22
				13:01

-	46	equijoin and (@ad<20010426)	USPAT;
			US-PGPUB;
			IBM_TDB
				2003/05/18
				07:51

-	0	(online with aggregat$3 with adapt$3 with (quer$5 or search$3)).ab. and (@ad<20010426)	USPAT;
			US-PGPUB;
			IBM_TDB
				2003/05/18
				07:58

-	1	(online with aggregat$3 with (quer$5 or search$3)).ab. and (@ad<20010426)	USPAT;
			US-PGPUB;
			IBM_TDB
				2003/05/18
				08:00

-	0	(online adj2 aggregat$3 adj2 (quer$5 or search$3)) and (@ad<20010426)	USPAT;
			US-PGPUB;
			IBM_TDB
				2003/05/18
				08:09

-	0	(parallel$3 with (hash adj ripple adj join)) and (@ad<20010426)	USPAT;
			US-PGPUB;
			IBM_TDB
				2003/05/18
				08:10

-	0	(hash adj ripple adj join) and (@ad<20010426)	USPAT;
			US-PGPUB;
			IBM_TDB
				2004/04/22
				13:02

-	6	(hash with join) same (partition$3 with tuple) and (@ad<20010426)	USPAT;
			US-PGPUB;
			IBM_TDB
				2003/05/18
				08:52

-	30	(hash$3 with (join$3 or merg$3 or combin$5)).ab. and (@ad<20010426) and partition$3	USPAT;
			US-PGPUB;
			IBM_TDB
				2003/11/14
				00:42

-	38	(((equivalent or hash$3) with (join$3 or merg$3 or combin$5)) with (processors or processers)) and partition$3 and (@ad<20010426)	USPAT;
			US-PGPUB;
			IBM_TDB
				2003/11/14
				14:22

-	10	(((equivalent or hash$3) with (join$3 or merg$3 or combin$5)) with (distribut$3 or redistribut$3) with (partition$3 or divid$3)) and (@ad<20010426)	USPAT;
			US-PGPUB;
			IBM_TDB
				2003/11/14
				09:39

IEEE HOME | SEARCH IEEE | SHOP | WEB ACCOUNT | CONTACT IEEE

IEEE

Membership Publications/Services Standards Conferences Careers/Jobs

IEEE Xplore®
RELEASE 1.6

Welcome
United States Patent and Trademark Office

IEEE Xplo
1 Million Doc
1 Million User

» Search Res

Help FAQ Terms IEEE Peer Review

Quick Links

Welcome to IEEE Xplore®

○ Home
○ What Can I Access?
○ Log-out

Tables of Contents

○ Journals & Magazines
○ Conference Proceedings
○ Standards

Search

○ By Author
○ Basic
○ Advanced

Member Services

○ Join IEEE
○ Establish IEEE Web Account
○ Access the IEEE Member Digital Library

Your search matched **68** of **1027552** documents.
A maximum of **500** results are displayed, **15** to a page, sorted by **Relevance** in **Descending** order.

Refine This Search:
You may refine your search by editing the current search expression or entering a new one in the text box.

(hash <sentence> join) Search

□ Check to search within this result set

Results Key:
JNL = Journal or Magazine **CNF** = Conference **STD** = Standard

1 **Parallel Star Join+DataIndexes: efficient query processing in data warehouses and OLAP**
Datta, A.; VanderMeer, D.; Ramamritham, K.;
Knowledge and Data Engineering, IEEE Transactions on , Volume: 14 , Issue: 6 , Nov.-Dec. 2002
Pages:1299 - 1316

[Abstract] [PDF Full-Text (1973 KB)] **IEEE JNL**

2 **Optimizing main-memory join on modern hardware**
Manegold, S.; Boncz, P.; Kersten, M.;
Knowledge and Data Engineering, IEEE Transactions on , Volume: 14 , Issue: 4 , July-Aug. 2002
Pages:709 - 730

[Abstract] [PDF Full-Text (3938 KB)] **IEEE JNL**

3 **Criss-cross hash joins: design and analysis**
Gopal, R.D.; Ramesh, R.; Zionts, S.;
Knowledge and Data Engineering, IEEE Transactions on , Volume: 13 , Issue: 4 , July-Aug. 2001
Pages:637 - 653

[Abstract] [PDF Full-Text (2448 KB)] **IEEE JNL**

4 **Parallel execution of hash joins in parallel databases**
Hui-I Hsiao; Ming-Syan Chen; Yu, P.S.;
Parallel and Distributed Systems, IEEE Transactions on , Volume: 8 , Issue: 8 , Aug. 1997

Pages:872 - 883

[Abstract] [PDF Full-Text (556 KB)] **IEEE JNL**

5 **A parallel distributive join algorithm for cube-connected multiprocessors**
Chung, S.M.; Jaerheen Yang;
Parallel and Distributed Systems, IEEE Transactions on , Volume: 7 , Issue: 2 , Feb. 1996
Pages:127 - 137

[Abstract] [PDF Full-Text (992 KB)] **IEEE JNL**

6 **Utilizing page-level join index for optimization in parallel join execution**
Chiang Lee; Zue-An Chang;
Knowledge and Data Engineering, IEEE Transactions on , Volume: 7 , Issue: 6 , Dec. 1995
Pages:900 - 914

[Abstract] [PDF Full-Text (1648 KB)] **IEEE JNL**

7 **Dynamic load balancing in multicomputer database systems using partition tuning**
Hua, K.A.; Chiang Lee; Hua, C.M.;
Knowledge and Data Engineering, IEEE Transactions on , Volume: 7 , Issue: 6 , Dec. 1995
Pages:968 - 983

[Abstract] [PDF Full-Text (1644 KB)] **IEEE JNL**

8 **Distributed load balancing for parallel main memory hash join**
Tout, W.R.; Praminik, S.;
Parallel and Distributed Systems, IEEE Transactions on , Volume: 6 , Issue: 8 , Aug. 1995
Pages:841 - 849

[Abstract] [PDF Full-Text (824 KB)] **IEEE JNL**

9 **Applying segmented right-deep trees to pipelining multiple hash joins**
Ming-Syan Chen; Mingling Lo; Yu, P.S.; Young, H.C.;
Knowledge and Data Engineering, IEEE Transactions on , Volume: 7 , Issue: 4 , Aug. 1995
Pages:656 - 668

[Abstract] [PDF Full-Text (1252 KB)] **IEEE JNL**

10 **Sort vs. hash revisited**
Graefe, G.; Linville, A.; Shapiro, L.D.;
Knowledge and Data Engineering, IEEE Transactions on , Volume: 6 , Issue: 6 , Dec. 1994
Pages:934 - 944

[Abstract] [PDF Full-Text (1100 KB)] **IEEE JNL**

11 **Parallel hash-based join algorithms for a shared-everything environment**
Martin, T.P.; Larson, P.-A.; Deshpande, V.;
Knowledge and Data Engineering, IEEE Transactions on , Volume: 6 , Issue: 5 , Oct. 1994
Pages:750 - 763

[Abstract] [PDF Full-Text (1156 KB)] **IEEE JNL**

12 **A parallel hash join algorithm for managing data skew**
Wolf, J.L.; Yu, P.S.; Turek, J.; Dias, D.M.;
Parallel and Distributed Systems, IEEE Transactions on , Volume: 4 , Issue: 12 , Dec. 1993
Pages:1355 - 1371

[Abstract] [PDF Full-Text (1636 KB)] **IEEE JNL**

13 **The adaptive-hash join algorithm for a hypercube multicomputer**
Omiecinski, E.; Lin, E.T.;
Parallel and Distributed Systems, IEEE Transactions on , Volume: 3 , Issue: 3 , May 1992
Pages:334 - 349

[Abstract] [PDF Full-Text (1180 KB)] **IEEE JNL**

14 **Comments on `Hash-based and index-based join algorithms for cube and ring connected multicomputers' by E.R. Omiecinski and E.T. Lin**
Tien, J.-Y.; Yang, W.-P.;
Knowledge and Data Engineering, IEEE Transactions on , Volume: 3 , Issue: 3 , Sept. 1991
Pages:387 - 389

[Abstract] [PDF Full-Text (312 KB)] **IEEE JNL**

15 **Heap-filter merge join: a new algorithm for joining medium-size inputs**
Graefe, G.;
Software Engineering, IEEE Transactions on , Volume: 17 , Issue: 9 , Sept. 1991
Pages:979 - 982

[Abstract] [PDF Full-Text (336 KB)] **IEEE JNL**

1 2 3 4 5 **Next**

Home | Log-out | Journals | Conference Proceedings | Standards | Search by Author | Basic Search | Advanced Search | Join IEEE | Web Account | New this week | OPAC Linking Information | Your Feedback | Technical Support | Email Alerting | No Robots Please | Release Notes | IEEE Online Publications | Help | FAQ| Terms | Back to Top

Welcome to IEEE Xplore®

- Home
- What Can I Access?
- Log-out

Tables of Contents

- Journals & Magazines
- Conference Proceedings
- Standards

Search

- By Author
- Basic
- Advanced

Member Services

- Join IEEE
- Establish IEEE Web Account
- Access the IEEE Member Digital Library

Your search matched **0** of **1027552** documents.
A maximum of **500** results are displayed, **15** to a page, sorted by **Relevance** in **Descending** order.

Refine This Search:
You may refine your search by editing the current search expression or entering a new one in the text box.

(hash <sentence> join) and (split <sentence> vector Search

□ Check to search within this result set

Results Key:
JNL = Journal or Magazine **CNF** = Conference **STD** = Standard

Results:
No documents matched your query.

Home | Log-out | Journals | Conference Proceedings | Standards | Search by Author | Basic Search | Advanced Search | Join IEEE | Web Account | New this week | OPAC Linking Information | Your Feedback | Technical Support | Email Alerting | No Robots Please | Release Notes | IEEE Online Publications | Help | FAQ| Terms | Back to Top

PORTAL
US Patent & Trademark Office

Subscribe (Full Service) Register (Limited Service, Free) Login

Search: ● The ACM Digital Library ○ The Guide

"hash join" and "split vector"

THE ACM DIGITAL LIBRARY

Feedback Report a problem Satisfaction survey

Terms used **hash join** and **split vector**

Found **173** of **131,734**

Sort results by: relevance

Display results: expanded form

Save results to a Binder

Search Tips

□ Open results in a new window

Try an Advanced Search

Try this search in The ACM Guide

Results 1 - 20 of 173

Result page: **1** 2 3 4 5 6 7 8 9 next

Relevance scale □ ▭ ▬ ▬ ■

1 Research sessions: query processing I: A scalable hash ripple join algorithm

Gang Luo, Curt J. Ellmann, Peter J. Haas, Jeffrey F. Naughton

June 2002 **Proceedings of the 2002 ACM SIGMOD international conference on Management of data**

Full text available: pdf(1.12 MB) Additional Information: full citation, abstract, references, citings, index terms

Recently, Haas and Hellerstein proposed the hash ripple join algorithm in the context of online aggregation. Although the algorithm rapidly gives a good estimate for many join-aggregate problem instances, the convergence can be slow if the number of tuples that satisfy the join predicate is small or if there are many groups in the output. Furthermore, if memory overflows (for example, because the user allows the algorithm to run to completion for an exact answer), the algorithm degenerates to bl ...

2 TID hash joins

Robert Marek, Erhard Rahm

November 1994 **Proceedings of the third international conference on Information and knowledge management**

Full text available: pdf(1.13 MB) Additional Information: full citation, references, index terms

3 Partially preemptible hash joins

Hwee Hwa Pang, Michael J. Carey, Miron Livny

June 1993 **ACM SIGMOD Record , Proceedings of the 1993 ACM SIGMOD international conference on Management of data**, Volume 22 Issue 2

Full text available: pdf(1.42 MB) Additional Information: full citation, abstract, references, citings, index terms

With the advent of real-time and goal-oriented database systems, priority scheduling is likely to be an important feature in future database management systems. A consequence of priority scheduling is that a transaction may lose its buffers to higher-priority transactions, and may be given additional memory when transactions leave the system. Due to their heavy reliance on main memory, hash joins are especially vulnerable to fluctuations in memory availability. Previous studies have propose ...

4 Spatial hash-joins

Ming-Ling Lo, Chinya V. Ravishankar

June 1996 **ACM SIGMOD Record , Proceedings of the 1996 ACM SIGMOD international conference on Management of data**, Volume 25 Issue 2

Full text available: pdf(1.35 MB) Additional Information: full citation, abstract, references, citings, index terms

We examine how to apply the hash-join paradigm to spatial joins, and define a new framework for spatial hash-joins. Our spatial partition functions have two components: a set of bucket extents and an assignment function, which may map a data item into multiple buckets. Furthermore, the partition functions for the two input datasets may be different.We have designed and tested a spatial hash-join method based on this framework. The partition function for the inner dataset is initialized by sampli ...

5 On parallel execution of multiple pipelined hash joins □

Hui-I Hsiao, Ming-Syan Chen, Philip S. Yu

May 1994 **ACM SIGMOD Record , Proceedings of the 1994 ACM SIGMOD international conference on Management of data**, Volume 23 Issue 2

Full text available: pdf(1.24 MB) Additional Information: full citation, abstract, references, citings, index terms

In this paper we study parallel execution of multiple pipelined hash joins. Specifically, we deal with two issues, processor allocation and the use of hash filters, to improve parallel execution of hash joins. We first present a scheme to transform a bushy execution tree to an allocation tree, where each node denotes a pipeline. Then, processors are allocated to the nodes in the allocation tree based on the concept of synchronous execution time such that inner relations (i.e., hash tables) ...

6 On optimal processor allocation to support pipelined hash joins □

Ming-Ling Lo, Ming-Syan Syan Chen, C. V. Ravishankar, Philip S. Yu

June 1993 **ACM SIGMOD Record , Proceedings of the 1993 ACM SIGMOD international conference on Management of data**, Volume 22 Issue 2

Full text available: pdf(994.72 KB) Additional Information: full citation, abstract, references, citings, index terms

In this paper, we develop algorithms to achieve optimal processor allocation for pipelined hash joins in a multiprocessor-based database system. A pipeline of hash joins is composed of several stages, each of which is associated with one join operation. The whole pipeline is executed in two phases: (1) the table-building phase, and (2) the tuple-probing phase. We focus on the problem of allocating processors to the stages of a pipeline to minimize the query execution time. We formulate the ...

7 Accurate modeling of the hybrid hash join algorithm □

Jignesh M. Patel, Michael J. Carey, Mary K. Vernon

May 1994 **ACM SIGMETRICS Performance Evaluation Review , Proceedings of the 1994 ACM SIGMETRICS conference on Measurement and modeling of computer systems**, Volume 22 Issue 1

Full text available: pdf(1.38 MB) Additional Information: full citation, abstract, references, citings, index terms

The join of two relations is an important operation in database systems. It occurs frequently in relational queries, and join performance is a significant factor in overall system performance. Cost models for join algorithms are used by query optimizers to choose efficient query execution strategies. This paper presents an efficient analytical model of an important join method, the hybrid hash join algorithm, that captures several key features of the algorithm's performance—including ...

8 Application of domain vector perfect hash join for multimedia data mining □

Venkata N. Rao Goli, William Perrizo

April 1997 **Proceedings of the 1997 ACM symposium on Applied computing**

Full text available: pdf(581.74 KB) Additional Information: full citation, references, index terms

Keywords: configurators, data warehouse, decision support systems, domain vectors, join processing, n-dimensional bit vectors, query vectors

9 Join algorithm costs revisited □
Evan P. Harris, Kotagiri Ramamohanarao
January 1996 **The VLDB Journal — The International Journal on Very Large Data Bases**, Volume 5 Issue 1
Full text available: pdf(329.00 KB) Additional Information: full citation, abstract, citings, index terms

A method of analysing join algorithms based upon the time required to access, transfer and perform the relevant CPU-based operations on a disk page is proposed. The costs of variations of several of the standard join algorithms, including nested block, sort-merge, GRACE hash and hybrid hash, are presented. For a given total buffer size, the cost of these join algorithms depends on the parts of the buffer allocated for each purpose. For example, when joining two relations using the nested block j ...

Keywords: Join algorithms, Minimisation, Optimal buffer allocation

10 Query evaluation techniques for large databases □
Goetz Graefe
June 1993 **ACM Computing Surveys (CSUR)**, Volume 25 Issue 2
Full text available: pdf(9.37 MB) Additional Information: full citation, abstract, references, citings, index terms, review

Database management systems will continue to manage large data volumes. Thus, efficient algorithms for accessing and manipulating large sets and sequences will be required to provide acceptable performance. The advent of object-oriented and extensible database systems will not solve this problem. On the contrary, modern data models exacerbate the problem: In order to manipulate large sets of complex objects as efficiently as today's database systems manipulate simple records, query-processi ...

Keywords: complex query evaluation plans, dynamic query evaluation plans, extensible database systems, iterators, object-oriented database systems, operator model of parallelization, parallel algorithms, relational database systems, set-matching algorithms, sort-hash duality

11 Exploiting early sorting and early partitioning for decision support query processing □
J. Claussen, A. Kemper, D. Kossmann, C. Wiesner
December 2000 **The VLDB Journal — The International Journal on Very Large Data Bases**, Volume 9 Issue 3
Full text available: pdf(478.23 KB) Additional Information: full citation, abstract, index terms

Decision support queries typically involve several joins, a grouping with aggregation, and/or sorting of the result tuples. We propose two new classes of query evaluation algorithms that can be used to speed up the execution of such queries. The algorithms are based on (1) *early sorting* and (2) *early partitioning*- or a combination of both. The idea is to push the sorting and/or the partitioning to the leaves, i.e., the base relations, of the query evaluation plans (QEPs) and ...

Keywords: Decision Support Systems, Early sorting and partitioning, Hash joins and hash teams, Performance evaluation, Query processing and optimization

12 Join processing in relational databases □
Priti Mishra, Margaret H. Eich
March 1992 **ACM Computing Surveys (CSUR)**, Volume 24 Issue 1
Full text available: pdf(4.42 MB)   Additional Information: full citation, abstract, references, citings, index terms, review

The join operation is one of the fundamental relational database query operations. It facilitates the retrieval of information from two different relations based on a Cartesian product of the two relations. The join is one of the most diffidult operations to implement efficiently, as no predefined links between relations are required to exist (as they are with network and hierarchical systems). The join is the only relational algebra operation that allows the combining of related tuples fro ...

Keywords: database machines, distributed processing, join, parallel processing, relational algebra

13 Optimizing database architecture for the new bottleneck: memory access □
Stefan Manegold, Peter A. Boncz, Martin L. Kersten
December 2000 **The VLDB Journal — The International Journal on Very Large Data Bases**, Volume 9 Issue 3
Full text available: pdf(357.33 KB)   Additional Information: full citation, abstract, index terms

In the past decade, advances in the speed of commodity CPUs have far out-paced advances in memory latency. Main-memory access is therefore increasingly a performance bottleneck for many computer applications, including database systems. In this article, we use a simple scan test to show the severe impact of this bottleneck. The insights gained are translated into guidelines for database architecture, in terms of both data structures and algorithms. We discuss how vertically fragmented data struc ...

Keywords: Decomposed storage model, Implementation techniques, Join algorithms, Main-memory databases, Memory access optimization, Query processing

14 Fast joins using join indices □
Zhe Li, Kenneth A. Ross
April 1999 **The VLDB Journal — The International Journal on Very Large Data Bases**, Volume 8 Issue 1
Full text available: pdf(263.06 KB)   Additional Information: full citation, abstract, index terms

Two new algorithms, "Jive join" and "Slam join," are proposed for computing the join of two relations using a join index. The algorithms are duals: Jive join range-partitions input relation tuple ids and then processes each partition, while Slam join forms ordered runs of input relation tuple ids and then merges the results. Both algorithms make a single sequential pass through each input relation, in addition to one pass through the join index and two passes through a te ...

Keywords: Decision support systems, Query processing

15 Buffer management based on return on consumption in a multi-query environment □
Philip S. Yu, Douglas W. Cornell
January 1993 **The VLDB Journal — The International Journal on Very Large Data Bases**,

Volume 2 Issue 1
Full text available: pdf(1.69 MB) Additional Information: full citation, abstract, references

In a multi-query environment, the marginal utilities of allocating additional buffer to the various queries can be vastly different. The conventional approach examines each query in isolation to determine the optimal access plan and the corresponding locality set. This can lead to performance that is far from optimal. As each query can have different access plans with dissimilar locality sets and sensitivities to memory requirement, we employ the concepts of memory consumption and return on cons ...

Keywords: buffer management, join methods, query optimization, queueing model, simulated annealing, simulation

16 Sing the truth about *ad hoc* join costs □
Laura M. Haas, Michael J. Carey, Miron Livny, Amit Shukla
August 1997 **The VLDB Journal — The International Journal on Very Large Data Bases**, Volume 6 Issue 3
Full text available: pdf(349.97 KB) Additional Information: full citation, abstract, index terms

In this paper, we re-examine the results of prior work on methods for computing *ad hoc* joins. We develop a detailed cost model for predicting join algorithm performance, and we use the model to develop cost formulas for the major *ad hoc* join methods found in the relational database literature. We show that various pieces of "common wisdom" about join algorithm performance fail to hold up when analyzed carefully, and we use our detailed cost model to derive op timal buff ...

Keywords: Buffer allocation, Cost models, Join methods, Optimization, Performance

17 Domain vector hashing for earth system data querying □
William Perrizo, Venkata Nagarjuna Rao Goli
February 1995 **Proceedings of the 1995 ACM symposium on Applied computing**
Full text available: pdf(629.35 KB) Additional Information: full citation, references, citings, index terms

Keywords: database, hash, join, query

18 Managing memory for real-time queries □
Hwee Hwa Pang, Michael J. Carey, Miron Livny
May 1994 **ACM SIGMOD Record , Proceedings of the 1994 ACM SIGMOD international conference on Management of data**, Volume 23 Issue 2
Full text available: pdf(1.59 MB) Additional Information: full citation, abstract, references, citings, index terms

The demanding performance objectives that real-time database systems (RTDBS) face necessitate the use of priority resource scheduling. This paper introduces a Priority Memory Management (PMM) algorithm that is designed to schedule queries in RTDBS. PMM attempts to minimize the number of missed deadlines by adapting both its multiprogramming level and its memory allocation strategy to the characteristics of the offered workload. A series of simulation experiments confirms th ...

19 An adaptive query execution system for data integration □
Zachary G. Ives, Daniela Florescu, Marc Friedman, Alon Levy, Daniel S. Weld
June 1999 **ACM SIGMOD Record , Proceedings of the 1999 ACM SIGMOD international**

conference on Management of data, Volume 28 Issue 2

Full text available: pdf(1.59 MB) Additional Information: full citation, abstract, references, citings, index terms

Query processing in data integration occurs over network-bound, autonomous data sources. This requires extensions to traditional optimization and execution techniques for three reasons: there is an absence of quality statistics about the data, data transfer rates are unpredictable and bursty, and slow or unavailable data sources can often be replaced by overlapping or mirrored sources. This paper presents the Tukwila data integration system, designed to support adaptivity at its core using ...

20 Join processing in database systems with large main memories
Leonard D. Shapiro
August 1986 **ACM Transactions on Database Systems (TODS)**, Volume 11 Issue 3

Full text available: pdf(1.41 MB) Additional Information: full citation, abstract, references, citings, index terms, review

We study algorithms for computing the equijoin of two relations in a system with a standard architecture hut with large amounts of main memory. Our algorithms are especially efficient when the main memory available is a significant fraction of the size of one of the relations to he joined; but they can be applied whenever there is memory equal to approximately the square root of the size of one relation. We present a new algorithm which is a hybrid of two hash-based algorithms and which dom ...

Results 1 - 20 of 173 Result page: **1** 2 3 4 5 6 7 8 9 next

L Number	Hits	Search Text	DB	Time stamp

1	126	"HASH JOIN" OR "HASH JOINS" OR "HASH JOINING"	USPAT;
			US-PGPUB;
			EPO;
			IBM_TDB
				2004/04/22 15:50

2	97	("HASH JOIN" OR "HASH JOINS" OR "HASH JOINING") AND @AD<20010426	USPAT;
			US-PGPUB;
			EPO;
			IBM_TDB
				2004/04/22 15:51

3	63	(("HASH JOIN" OR "HASH JOINS" OR "HASH JOINING") AND @AD<20010426) AND PARTITION$3	USPAT;
			US-PGPUB;
			EPO;
			IBM_TDB
				2004/04/22 15:52

4	8	((("HASH JOIN" OR "HASH JOINS" OR "HASH JOINING") AND @AD<20010426) AND PARTITION$3) AND (PARTITION$3 WITH NODE)	USPAT;
			US-PGPUB;
			EPO;
			IBM_TDB
				2004/04/22 15:57

5	0	842991.APN.	USPAT;
			US-PGPUB;
			EPO;
			IBM_TDB
				2004/04/22 16:01

6	49	("HASH JOIN" OR "HASH JOINS" OR "HASH JOINING") AND (INTERMEDIATE WITH RESULTS)	USPAT;
			US-PGPUB;
			EPO;
			IBM_TDB
				2004/04/22 16:03

7	65	("HASH JOIN" OR "HASH JOINS" OR "HASH JOINING") AND (DISPLAY WITHINTERMEDIATE WITH RESULTS)	USPAT;
			US-PGPUB;
			EPO;
			IBM_TDB
				2004/04/22 16:03

8	0	("HASH JOIN" OR "HASH JOINS" OR "HASH JOINING") AND (DISPLAY WITH INTERMEDIATE WITH RESULTS)	USPAT;
			US-PGPUB;
			EPO;
			IBM_TDB
				2004/04/22 16:03

9	0	(("HASH JOIN" OR "HASH JOINS" OR "HASH JOINING") AND (INTERMEDIATE WITH RESULTS)) AND (DISPLAY WITH INTERMEDIATE)	USPAT;
			US-PGPUB;
			EPO;
			IBM_TDB
				2004/04/22 16:04

10	0	("HASH JOIN" OR "HASH JOINS" OR "HASH JOINING") AND (DISPLAY WITH INTERMEDIATE)	USPAT;
			US-PGPUB;
			EPO;
			IBM_TDB
				2004/04/22 16:06

11	9199	(DISPLAY WITH INTERMEDIATE)	USPAT;
			US-PGPUB;
			EPO;
			IBM_TDB
				2004/04/22 16:06

12	365	JOIN AND (DISPLAY WITH INTERMEDIATE)	USPAT;
			US-PGPUB;
			EPO;
			IBM_TDB
				2004/04/22 16:06

13	166	(JOIN AND (DISPLAY WITH INTERMEDIATE)) AND DISTRIBUT$3	USPAT;
			US-PGPUB;
			EPO;
			IBM_TDB
				2004/04/22 16:07

14	38	((JOIN AND (DISPLAY WITH INTERMEDIATE)) AND DISTRIBUT$3) AND NODE	USPAT;
			US-PGPUB;
			EPO;
			IBM_TDB
				2004/04/22 16:08

15	28	(((JOIN AND (DISPLAY WITH INTERMEDIATE)) AND DISTRIBUT$3) AND NODE) AND @AD<20010426	USPAT;
			US-PGPUB;
			EPO;
			IBM_TDB
				2004/04/22 16:08

	U	1	Document ID	Issue Date	Pages	Title	Current OR
1	☒	□	US 6625593 B1	20030923	13	Parallel query optimization strategies for replicated and partitioned tables	707/2
2	☒	□	US 6341281 B1	20020122	27	Database system with methods for optimizing performance of correlated subqueries by reusing invariant results of operator tree	707/3
3	☒	□	US 5970490 A	19991019	31	Integration platform for heterogeneous databases	707/10
4	☒	□	US 5864842 A	19990126	15	Optimization of SQL queries using hash star join operations	707/3
5	☒	□	US 5758345 A	19980526	23	Program and method for establishing a physical database layout on a distributed processor system	707/100
6	☒	□	US 5557791 A	19960917	27	Outer join operations using responsibility regions assigned to inner tables in a relational database	707/2
7	☒	□	US 5551031 A	19960827	26	Program storage device and computer program product for outer join operations using responsibility regions assigned to inner tables in a relational database	707/2
8	☒	□	US 5379419 A	19950103	54	Methods and apparatus for accesssing non-relational data files using relational queries	707/4

	Current XRef	Retrieval Classif	Inventor	S	C	P	2	3	4	5
1	707/4		Leung, Ting Yu et al.	□	□	□	□	□	□	□
2	707/2; 707/4		MacNicol, Roger Dermot et al.	□	□	□	□	□	□	□
3	707/104.1		Morgenstern, Matthew	□	□	□	□	□	□	□
4	707/2		Pederson, Donald Raymond et al.	□	□	□	□	□	□	□

5	707/200;
	707/206;
	711/1;
	711/5
			Wang, James Chien	□	□	□	□	□	□	□

6			Cheng, Josephine M. et al.	□	□	□	□	□	□	□
7			M. Cheng, Josephine et al.	□	□	□	□	□	□	□
8	707/2		Heffernan, John S. et al.	□	□	□	□	□	□	□

	Image Doc. Displayed	PT
1	US 6625593	□
2	US 6341281	□
3	US 5970490	□
4	US 5864842	□
5	US 5758345	□
6	US 5557791	□
7	US 5551031	□
8	US 5379419	□